REMARKS

Claims 1-35 are currently pending in the subject application and are presently under consideration. Claim 18 has been amended to render language in the body to be consistent with language in the preamble, and is not intended to narrow the scope of such claim. Claims 27 and 30 have been amended to clarify operability of the invention. All pending claims with status identifiers are listed at pages 2-7.

Applicants' representative acknowledges with appreciation the Examiner's indication that claims 10-17 and 22 are allowed; and objected to claims 9 and 19-21 would be allowed if recast in independent form to include all limitations of the respective base claim and any intervening claims. However, it is believed such amendments are not necessary in view of the deficiencies discussed *infra* of the cited art *vis a vis* applicants' claimed invention.

Favorable reconsideration of the subject patent application is respectfully requested in view of the comments herein.

I. Rejection of Claims 1-8, 18, and 23-35 Under 35 U.S.C. § 103(a)

Claims 1-8, 18, and 23-35 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Evans, *et al.* (US 5,805,899) in view of Buxton (US 6,182,279). It is respectfully submitted that this rejection be withdrawn for at least the following reasons. Neither Evans, *et al.* nor Buxton, individually or in combination, teach or suggest every limitation of the applicants' invention as recited in the subject claims.

> To reject claims in an application under §103, an examiner must establish a *prima facie* case of obviousness. A *prima facie* case of obviousness is established by a showing of three basic criteria. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) ***must teach or suggest all the claim limitations***. *See* MPEP §706.02(j). The ***teaching or suggestion to make the claimed combination*** and the

> reasonable expectation of success ***must both be found in the prior art and not based on applicant's disclosure.*** *See In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991) (emphasis added).

In particular, regarding independent claims 1, 18, and 23, neither Evans, *et al.* nor Buxton teach or suggest ***an assembly with a manifest that contains a list of modules that make up the assembly***, and thus cannot teach or suggest ***providing the manifest with a hash of the contents of at least one module of the list of modules*** as recited in the subject claims and defined in the specification. The subject invention facilitates ensuring integrity of assemblies necessary for proper operation of a software application at runtime *via* utilizing a hash to ascertaining whether contents of modules that make up the assembly have been modified. As is known in the art, an assembly refers to a grouping of files (modules) necessary to perform a particular application, and modules are portion(s) of a computer program that are created to carry out a particular function within the application, and can be utilized alone or combined with other modules in connection with enabling proper operation of the particular application. Thus, for example, an assembly as recited in the subject claims can comprise a dynamic linked library (DLL) containing one or modules, a manifest that contains a list of the modules, and a hash of one or more of the modules in the DLL. Assemblies (*e.g.*, DLLs, executables, ...) facilitate reduction of code stored in memory, as a single block of code can be shared by a plurality of application tasks rather than the plurality of tasks each containing copies of modules that are required for proper operation of the task. The file structure (*e.g.*, providing a manifest with a list of module(s) in an assembly) combined with providing a hash of contents of the module(s) mitigates errors that can occur during instances that a first application task modifies a module that is utilized by a second application task. Conventional systems and/or methods are unable to determine existence of such modifications, thereby resulting in inoperability of the second application task and possibly causing costly hardware and/or software damage. Moreover, the present invention as recited in these claims can discover modifications to modules by untrusted

sources, and therefore mitigate damage to computer software and hardware that can occur, for example, *via* a computer virus.

In contrast, Evans, *et al.* discloses a system and method for generating a versioned object at build time, and thereafter determining if all versioned objects necessary for operation are present at runtime (*See* abst.). The shared object is created *via* inputting a mapfile and a compiled object that is to become a shared object into a link editor. The mapfile specifies the global symbols and a version name for each version of the shared object (*See* Fig. 2, col. 4 lines 65-67, and col. 5 lines 1-6). The shared object is created in an ELF format, and comprises various sections (*See* Fig. 5). For instance, the shared object contains a version definition section 506, a version symbol section 508, and a version dependency section 510. However, the sections in the shared object are not ***modules*** as recited in the subject claims, as the sections are informational and cannot carry out a particular function. The Examiner contends that such sections represent individual objects, and are therefore equivalent to a list of modules. However, as the sections cannot carry out particular functions, they cannot be construed as ***modules*** as recited in these claims, and therefore cannot be deemed *equivalent* to a list of modules.

Evans, *et al.* further discloses creating a dynamic executable *via* inputting the shared object and relocatable object code into a link-editor (*See* Fig. 2(b), col. 5 lines 7-13). The dynamic executable has a file format similar to a created shared object, wherein the file comprises a plurality of sections, each section associated with a version dependency section (*See* Fig. 10, col. 10 lines 32-51). Like the shared object, the dynamic executable disclosed in Evans, *et al.* does not contain ***modules***. Thus the sections contained in the dynamic executable cannot be equivalent to a ***list of modules*** as recited in the subject claims. Moreover, as Evans, *et al.* does not teach or suggest ***a list of modules***, Evans, *et al.* cannot teach or suggest ***a hash of the contents of at least one... of the modules***. Rather, Evans, *et al.* discloses utilizing a hash for version names of each version of the versioned object that is contained in the version definition section (*See* Fig. 6, col. 9 lines 35-37). As the hash is associated only with a version name and no underlying data, the hash is not a ***hash of the contents of a module***.

Buxton discloses a system that enables a first user to customize a base application, and thereafter relay such modifications in template form to a second user having an identical base application, wherein the second user can thereafter customize the base application according to the template. Like Evans, *et al.*, Buxton does not teach or suggest ***an assembly containing... a list of modules***, and therefore cannot teach ***a hash of the contents of at least... one of the modules.*** Rather, Buxton teaches a digital signature that identifies the component and the vendor or source of a component. However, Buxton does not teach or suggest ***listing*** such components in a ***manifest*** to facilitate ensuring integrity of ***an assembly that comprises*** the components.

Regarding independent claims 27 and 30, such claims have been amended to clarify operability of the subject invention as recited in these claims. Creating a hash of ***a manifest of the referenced assembly*** facilitates ensuring integrity of data within the assembly while minimizing computational cost required to ensure such integrity. For example, a hash of one or modules in the referenced assembly can be previously obtained – thus providing a hash of the manifest listing such modules provides for a convenient manner for ensuring integrity of the referenced assembly. Neither Evans, *et al.* nor Buxton teach or suggest the effective mechanism for ensuring integrity as recited in these claims. Evans, *et al.* teaches a section in a shared object that references one or more *versions* necessary for proper operation of an application task (*See* version dependency section of Figs. 10 and 11, col. 11 lines 9-25). Such version dependency section is created by the link-editor during instances that an application program references a global symbol in the interface to a shared object. Thus, if a program requires a particular global symbol, versions of a shared object will be searched for that particular global symbol, and versions comprising such symbol will be referenced. However, creating such a version dependency list and providing a hash of the names of such dependent versions does not make obvious the subject invention as recited in these claims, as ***a hash of a manifest of a referenced assembly*** is not provided. Like Evans, *et al.*, Buxton does not teach or suggest providing a hash of a manifest of a referenced assembly to facilitate ensuring integrity of the referenced assembly.

In view of at least the above, it is respectfully submitted that the rejection of

independent claims 1, 18, 23, 27, and 30 and dependent claims 2-8, 24-26, 28, 29, and 31-35, which respectively depend therefrom, be withdrawn.

II. Conclusion

The present application is believed to be condition for allowance in view of the above comments. A prompt action to such end is earnestly solicited.

In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063.

Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number listed below.

Respectfully submitted,
AMIN & TUROCY, LLP

Himanshu S. Amin
Reg. No. 40,894

AMIN & TUROCY, LLP
24TH Floor, National City Center
1900 E. 9TH Street
Cleveland, Ohio 44114
Telephone (216) 696-8730
Facsimile (216) 696-8731